Improving Query Results with Automatic Duplicate Detection

Irina Astrova

Institute of Cybernetics, Tallinn University of Technology, Akadeemia tee 21,
12618 Tallinn, Estonia
irinaastrova@yahoo.com

Abstract. Ontology-based data integration poses significant challenges. One is that an ontology used as a global reference model during the ontology-based data integration can contain duplicated attributes, which can easily lead to improper query results. This problem arises when merging similar or overlapping information from ontologies extracted from distributed digital libraries into a single global ontology. To solve the problem, we propose a novel context-based approach that analyzes a workload of queries over the single global ontology to automatically calculate (semantic) distances between attributes, which are then used for duplicate detection. We present experimental results to demonstrate the quality of our approach.

Keywords: Ontology-based data integration, duplicate detection, market basket analysis, ICD algorithm.

1 Introduction

Semantic heterogeneity is the ambiguous interpretation of terms describing the meaning of data in heterogeneous resources such as distributed digital libraries [1]. This is a well-known problem in data integration. A recent solution to this problem is to use ontologies; this is called *ontology-based data integration* [2, 3, 4].

There are three main advantages in ontology-based data integration:

1. Ontologies provide a rich-predefined vocabulary that serves as a uniform interface for querying over distributed digital libraries, the interface that is independent of the underlying database schemata.
2. The knowledge represented by ontologies is sufficiently comprehensive to support translation of all relevant data.
3. Ontologies support consistent management and recognition of inconsistent data [3, 5].

The main problem with ontology-based data integration is that ontologies are heterogeneous themselves [6]. In particular, they can be expressed in different ontology languages, each with its own syntax, semantics and expressivity. Even using the same ontology language does not solve the semantic heterogeneity problem, because ontologies can contain duplicated attributes.

As an example, consider a user who submits the following query against the Wikipedia infobox ontology [7]: "Which performers were born in Chicago?" In response to this query, the query-answering system will return only one result (viz. Michael Ian Black). However, if it were known that *actor* and *comedian* are subclasses of *performer* and that their attributes *birthplace*, *birth place*, *city of birth*, *place of birth* and *origin* are duplicates of *performer*'s *location*, the query-answering system could return 163 additional results. Thus, the recall of query results can be greatly improved by detecting duplicated attributes.

Duplicate detection may be manual, automatic, or both. Traditionally, duplicate detection is performed by humans (e.g. domain experts and users): "Humans do it better" [8]. Many ontology languages provide the means to specify duplicates. E.g. OWL [9] has a construct *sameAs*. However, humans cannot cope with a large set of attributes. As a result, manual duplicate detection tends to be slow, tedious and inefficient, and does not work on a large scale. Therefore, there is a need for automatic duplicate detection.

2 Related Work

Most research focuses on identifying similar attributes, with some research devoted to detecting duplicates. However, the approaches to identifying similar attributes are generally the same as those used to detect duplicates (i.e. very similar attributes). Over the past two decades, researchers in both academy and industry have proposed various approaches to identifying similar attributes. These approaches can be categorized as:

1. **Term-based (or linguistic) approaches** where *two attributes are considered to be similar if their names (i.e. terms) are similar* [8, 10].
2. **Value-based (or extensional) approaches** where *two attributes are considered to be similar if their values are similar* [8, 10].
3. **Structure-based (or taxonomic) approaches** where *two attributes are considered to be similar if their structures (i.e. taxonomies) are similar* [8, 10].
4. **Context-based approaches** where *two attributes are considered to be similar if their contexts are similar* [10].
5. **Hybrid approaches** that combine two or more approaches from the first four categories to minimize false positives (i.e. dissimilar attributes that appear similar) and false negatives (i.e. similar attributes that appear dissimilar) [8, 10].

The approaches can also be categorized based on the information they use for identifying similar attributes: domain knowledge from domain experts, the ontology itself (e.g. attribute names, values and taxonomies) and the past user interaction with the ontology (e.g. a workload of queries against the ontology and an edit history). E.g. Wu and Weld [7] proposed to use a history of changes made to the ontology and analyze this information to detect duplicates. There can be attributes in a class that are frequently renamed, or their values can be copied to one and the same attribute in another class. Such an edit history points to evidence that these attributes are

duplicates. However, the edit history must be recorded for a long time to minimize false positives and false negatives.

Furthermore, the approaches can also be categorized based on the techniques they use for identifying similar attributes: information retrieval, machine learning and graph theory techniques. E.g. Doan et al. [11], and Berlin and Motro [12] proposed to use machine learning techniques. These techniques require training data as input. In particular, users specify one or more positive examples, and the machine learning techniques attempt to retrieve similar attributes (based on the similarity of their values) through an iterative process of relevance feedback from the users. By contrast, Madhavan [13] proposed to use graph theory techniques for identifying similar attributes, which are organized in the taxonomy. These techniques assume that attributes appear to be similar if both those that precede them and those that succeed them do so.

3 Our Approach

A term-based approach can incur problems in situations where the same terms are used to name dissimilar attributes (i.e. homonyms) or where different terms are used to name similar attributes (i.e. synonyms). A value-based approach can incur problems in situations where similar attributes have no or few common values or where dissimilar attributes have many common values. A structure-based approach can incur problems in situations where similar attributes are not organized in the taxonomy or where the taxonomy is shallow. Since terms, values and structures are not sufficient criteria for identifying similar attributes, we propose to use a context-based approach where two attributes are considered to be similar if their contexts are similar. The main problem with this approach is how to identify similar contexts. We address this problem by adopting a similarity measure from market basket analysis.

Because domain experts are often unavailable and because the use of ontology itself for duplicate detection can be inefficient when the ontology contains many attributes, we propose to use the past user interaction with the ontology such as a workload of queries asked by users against the ontology. The workload typically contains a small set of attributes. Furthermore, the use of the workload does not require training data as input. This is a big advantage in the case where users submit queries against distributed digital libraries as training data are missing [14]. Moreover, gathering queries is much easier than gathering training data as profiling tools automatically record the queries into user profiles.

3.1 Market Basket Analysis

Market baskets are the sets of products bought together by customers in transactions. These may be the results of customer visits to the supermarket or customer online purchases in a virtual store. Typically, market baskets are represented as a binary matrix where rows correspond to transactions and columns to products. A row has a value of 1 for a column if the customer has bought the product in the transaction; otherwise, it is 0. The number of products and their price are ignored.

One of the most popular tasks of market basket analysis is to derive customer buying patterns. These patterns can be used to identify similar products. As an example, consider Coke and Pepsi. These two products appear dissimilar because they have few customers in common. However, it was observed that the customers of Coke and Pepsi bought many other products in common such as hamburgers, cheeseburgers, pizzas and chips. Based on this observation, Das and Mannila [15] defined the following similarity measure for products: *two products are considered to be similar if the buying patterns of their customers are similar*.

We adapt this similarity measure to attributes: *two attributes are considered to be similar if the querying patterns of their users are similar*. E.g. if it were known that there are many users who have asked about the birth place of actor together with the actor's name and birth date, and that there are many users who have asked about the origin of actor, again, together with the actor's name and birth date, we could conclude that attributes *birth place* and *origin* in a class *actor* are similar to each other.

We take advantage of relationship between querying patterns and user behaviors. In particular, users who have similar questions in mind submit similar queries, the queries with similar querying patterns [16]. E.g. when searching for a biography of actor, many users tend to ask about actor's name and birth date. Therefore, we propose to use a workload of queries asked by users against the ontology and analyze this information to derive querying patterns using data mining and pattern recognition techniques.

3.2 Assumptions

We assume that users do not ask about all attributes in the ontology at once. (This is by analogy with market basket analysis, which assumes that a market basket contains a small set of products from hundreds or thousands of products available in the supermarket or virtual store.) In the example above, the users have not asked about *actor*'s *nationality* and *marital status*. These are called *missing attributes*.

In addition, we assume that users understand the ontology well enough to submit queries that reveal the similarity between attributes, or the users intuitively know if the attributes are similar. E.g. there can be several recent queries in the workload by a certain user who may repeatedly ask about *actor*'s *birthplace*, *birth place*, *city of birth*, *place of birth* and *origin*.

3.3 Steps

Our approach goes through two basic steps:

1. Calculation of distances between attributes.
2. Detection of duplicates.

3.3.1 Calculation of Distances between Attributes

To calculate distances between attributes, we adopt the ICD (Iterated Contextual Distance) algorithm [15] from market basket analysis. The basic idea behind the ICD algorithm is to start with an arbitrary distance between attributes and use this distance to calculate a probability distribution of the attributes in the workload of queries, then use this distribution to recalculate the distance between the attributes. Since the calculation of a distance between attributes is circular, the ICD algorithm is iterative. A few iterations of the ICD algorithm (typically 5) produce a stable distance between attributes called an *iterated contextual distance*. This distance is between 0 and 1; 0 means that two attributes are completely similar and 1 means that they are completely dissimilar. Next, we present the ICD algorithm.

ICD ALGORITHM

INPUT: A workload of m queries over an ontology with n attributes.

OUTPUT: An n x n symmetric distance matrix in which an element standing in the i-th row and j-th column represents the iterated contextual distance between the attributes i and j.

1. **Construct a binary matrix.** Construct an m x n binary matrix *M* where rows correspond to the queries and columns to the attributes. Let *M*(i, j) be an element of the matrix *M* that stands in the i-th row and the j-th column. It has a value of 1 if the query i references the attribute j. Otherwise, it is 0.
2. **Construct a distance matrix.** Construct an n x n symmetric distance matrix *D* where both rows and columns correspond to the attributes. Let *D*(i, j) be an element of the matrix *D* that stands in the i-th row and the j-th column. It has a random value between 0 and 1 if $i \neq j$. Otherwise, it is 0.
3. **Construct query vectors.** Let R be a set of attributes in the ontology. For each attribute $A \in R$, let $r_A = \{t \mid M(t, A) = 1\}$ be a set of queries that reference the attribute A.
4. **Construct attribute vectors.** For each query $t \in r_A$, let $A_t = \{A \mid M(t, A) = 1\}$ be a set of attributes that the query t references.
5. **Construct probability distribution vectors**. For each attribute $A \in R$, let $V_A = \{f(t, A) \mid t \in r_A\}$ be its probability distribution vector, where f(t, A) is the probability distribution of the attribute A in the query t. It is calculated using formula (1):

$$f(t,A) = 1 - \prod_{A_i \in A_t} \left(1 - \frac{K(D(A_i,A))}{\sum_{C \in R} K(D(A_i,C))}\right) \quad (1)$$

   where K is a kernel smoothing function; e.g. K(X) = 1/(1+X).
6. **Calculate centroids of probability distribution vectors**. For each probability distribution vector V_A, let c_A be its centroid. It is calculated using formula (2):

$$c_A = \frac{1}{|V_A|} \sum_{f(t,A) \in V_A} f(t,A) \quad (2)$$

7. **Calculate distances between attributes.** For each pair of attributes $A \in R$ and $B \in R$ $(A \neq B)$, let $D(A, B) = D(B, A) = |c_A - c_B|$, where c_A and c_B are centroids of V_A and V_B, respectively.
8. **Iterate:** Stop if the algorithm converges. Otherwise, go to Step 5.

3.3.2 Detection of Duplicates

To detect duplicates, we use a threshold; e.g. 0.20. Any two attributes with the iterated contextual distance less than this threshold are considered to be duplicates. For each pair of attributes $A \in R$ and $B \in R$ $(A \neq B)$, let $S = \{(A, B) \mid D(A, B) < T\}$ be a set of duplicates, where $T \in [0, 1]$ is a threshold.

4 Experiments

We conducted experiments:

1. To evaluate the quality of the results produced by the ICD algorithm.
2. To find a relationship between this quality and the workload size.

4.1 Quality of Results Produced by the ICD Algorithm

In the experiments, we used a real ontology from the car sales domain [17]. This ontology did not require highly specialized domain knowledge, which simplified the task of finding people to participate in the experiments. Nor did the ontology contain many attributes, making it easy for the participants to query the ontology. In particular, the ontology had 17 attributes (both similar and dissimilar): *mileage, stock, stock number, color, miles, exterior, make, number of doors, body style, body type, drive wheels, horsepower, year, model, engine size, engine location,* and *price*.

We asked 2 people called "oracles" to rate each pair of attributes between 0 (completely similar) and 1 (completely dissimilar) using their intuition. These ratings constituted a gold standard.

In addition, we asked 5 people (different from the oracles) to provide us with queries that they would submit if they wanted to buy a car. Typically, the people asked about specific car characteristics in certain price range or explored various tradeoffs; e.g. price vs. horsepower. We collected a total of 37 queries, which constituted a workload. This workload was used as input to the ICD algorithm. When the ICD algorithm is run on the workload, it produced a binary matrix in Table 1 and a distance matrix in Table 2.

To evaluate the quality of the results produced by the ICD algorithm, we compared these results with the gold standard. The results appeared intuitive and reasonable. E.g. the ICD algorithm found the similarity between *miles* and *mileage* (a distance of 0.06), between *color* and *exterior* (a distance of 0.11), and between *stock* and *stock number* (a distance of 0.07).

Table 1. Binary matrix

	mileage	**stock**	**stock number**	**color**	**miles**	**exterior**	**…**	**price**
q1	1	0	0	1	1	0	…	1
q2	1	1	0	1	1	0	…	1
q3	1	0	1	0	0	1	…	1
q4	0	0	1	1	1	0	…	1
q5	0	1	0	0	0	1	…	1
q6	1	0	0	1	0	0	…	0
q7	0	0	0	1	1	0	…	1
q8	0	0	0	0	0	1	…	0
…	…	…	…	…	…	…	…	…
q37	0	1	0	0	0	1	…	1

Table 2. Distance matrix

	mileage	**stock**	**stock number**	**color**	**miles**	**exterior**	**…**	**price**
mileage	0.00	0.83	0.83	0.94	0.06	0.95	…	0.99
stock	0.83	0.00	0.07	0.93	0.83	0.95	…	0.98
stock number	0.83	0.07	0.00	0.93	0.83	0.95	…	0.98
color	0.94	0.93	0.93	0.00	0.94	0.11	…	0.92
miles	0.06	0.83	0.83	0.94	0.00	0.95	…	0.99
exterior	0.95	0.95	0.95	0.11	0.95	0.00	…	0.92
…	…	…	…	…	…	…	0.00	…
price	0.99	0.98	0.98	0.92	0.99	0.92	…	0.00

4.2 Quality vs. Workload Size

To find a relationship between the quality and the workload size, we ran the ICD algorithm on randomly sampled fractions of the workload. The experiments showed: the larger workload, the better quality.

5 Conclusion and Future work

We have proposed a novel context-based approach to automatically detecting duplicated attributes in an ontology, which adopts the ICD algorithm from market basket analysis. Our approach has been tested against the real ontology from the car sales domain.

Even though the ICD algorithm appears to converge quickly (typically within 5 iterations) in practice, criteria for that convergence are to be investigated. However, a theoretical analysis of the convergence is difficult, because the ICD algorithm essentially tries to compute fixed points of a non-linear dynamic system. Furthermore,

we'll investigate if some other approaches (such as the term, value and structure-based) can be combined with ours to produce even better results.

Acknowledgments. This work was supported by the Estonian Centre of Excellence in Computer Science (EXCS) funded mainly by the European Regional Development Fund (ERDF).

References

1. Adams, T., Dullea, J., Clark, P., Sripada, S., Barrett, T.: Semantic integration of heterogeneous information. In *CS&I'2000, 5th International Conference on Computer Science and Informatics* (2000)
2. Du, D., LePendu, P.: Ontology-based integration for relational databases. In *SAC'2006, Proceedings of the ACM Symposium on Applied Computing* (2006)
3. Buccella, A., Cechic, A., Brisaboa, N.: An ontology approach to data integration. *Journal of Computer Science and Technology*, Vol. 3, No.2, pp. 62-68 (2003)
4. Cui, Z., Jones, D., O'Brien, P.: Issues in ontology-based information integration. In *IJCAI'2001, 17th International Joint Conference on Artificial Intelligence* (2001)
5. Kruk S., Haslhofer B., Piotrowski P., Westerski A., Woroniecki T.: The role of ontologies in semantic digital libraries. In *NKOS'2006*, *5th European Networked Knowledge Organization Systems Workshop* (2006)
6. Klein, M.: Combining and relating ontologies: an analysis of problems and solutions. In *IJCAI'2001, Workshop Ontologies and Information Sharing at 17th International Joint Conference on Artificial Intelligence*, pp. 53-62 (2001)
7. Wu, F., Weld, D.: Automatically refining the Wikipedia infobox ontology. In *WWW'08, 17th International Conference on World Wide Web*. pp. 635-644 (2008)
8. Eyal, A., Gal, A., Jamil, H., Modica, H.: Automatic ontology matching using application semantics. *AI Magazine*, Vol. 26, issue 1, pp. 21-31 (2005)
9. OWL Web Ontology Language Reference, http://www.w3.org/TR/owl-ref (2004)
10. Ehrig, M., Haase, P., Hefke, M., Stojanovic, N.: Similarity for ontologies – a comprehensive framework. In *PAKM'2004, Workshop on Enterprise Modeling and Ontology: Ingredients for Interoperability* (2004)
11. Doan A., Madhavan J., Domingos P., Halevy A.: Learning to map between ontologies on the Semantic Web. In *IWWWC'2002, 11th International World Wide Web Conference*, ACM Press. pp. 662-673 (2002)
12. Berlin J., Motro A.: Autoplex: Automated Discovery of Content for Virtual. *Lecture Notes in Computer Science*, Springer. pp. 108-122 (2001)
13. Madhavan J., Bernstein P., Domingos P., Halevy A.: Representing and reasoning about mappings between domain models. In *AAAI'2002*, *11th National Conference on Artificial Intelligence*. pp. 80-86 (2002)
14. Renda, M., Straccia, U.: Automatic structured query transformation over distributed digital libraries. In *SAC'2006*, *ACM Symposium on Applied Computing*, pp. 1078-1083 (2006)
15. Das, G., Mannila, H.: Context-based similarity measures for categorical databases. In *PKDD'2000, 4th European Conference on Principles of Data Mining and Knowledge Discovery*. pp. 201-210 (2000)
16. Sapia C.: On modeling and predicting query behavior in OLAP systems, In *DMDW'1999*, *International Workshop on Design and Management of Data Warehouses*. pp. 1-10 (1999)
17. Astrova, I.: Toward the Semantic Web – an approach to reverse engineering of relational databases to ontologies. In *ADBIS'05, 9th East-European Conference on Advanced Databases and Information Systems*. pp. 111-122 (2005)